



Sketch Recognition

Many innovations begin as scribbles – but where once lots of brilliant brainwaves never left the drawing board, they can now be placed on a computer that turns them into working replicas. Jon Thompson considers how today's sketch recognition capabilities could assist the innovators of tomorrow

For generations of inventors, backs of envelopes and scraps of paper have been instrumental in the birth of new, often culture-changing ideas or inventions. Now, though, it looks as if researchers will replace the inspirational beer mat with software that can take a sketch drawn on a tablet PC's screen and construct a virtual model, with all the dynamics you would find in a physical prototype.

The next Thomas Edison or James Dyson could test ideas in hours instead of months. This same technology could help design and develop new and innovative products automatically, and turn the traditional student's notebook into the often-touted 'smart paper'.

GOOD ON PAPER

The idea of turning sketches on paper into working computer models goes back to the mid-1990s. In 1996, Mark Gross and Ellen Yi-Luen Do, then both at Colorado State University, started a project with the goal of making computer interaction as easy as using pen and paper. To do this, they decided to

find a way of making software interact on human terms. The technique, they decided, should also be usable as a general interface to other programs.

The problem with traditional user interfaces as Gross and Yi-Luen Do saw them was that they worked on the computer's terms and not the user's. For people used to working by sketching out ideas and alternatives on paper, such as engineers, existing design packages could become very frustrating to work with. The pair wanted to overcome that problem by making intelligent, digital paper. They came up with a program called the Electronic Cocktail Napkin. Written in Common Lisp for the Apple Macintosh, its main aim was to capture an engineer's experience directly, rather than have him or her explain it to the computer.

To do this, the Napkin must recognise what's being drawn. It uses a special algorithm to identify multi-line shapes. It records the path the pen takes in drawing a shape, the number of individual lines used, the position of any corners and curves, the size of the shape and even its aspect ratio. It

compares what the engineer draws with a large database of pre-programmed shapes. By performing rotations on the objects drawn, it can even recognise shapes that are inverted or set at an angle. Once recognised, the shapes remain as sketches on the screen, but the user can group them into ever larger and more complex structures. These in turn can be recognised by the Napkin, creating an environment driven by the needs of the user and not the computer.

The system also has a sketchbook that captures all previous sketches made by users, so it can learn from its own experience. The Napkin has a high level of understanding about the content of each sketch, so it's easy for it to sort them by conceptual similarity. Because of this, it's easy to find and take ideas from one design and use them in another similar one, and see how two designs are related.

By recognising shapes in any graphic presented to it, the Electronic Cocktail Napkin is also capable of interfacing with other systems in a unique way. The researchers have already demonstrated this ability by using the Napkin as the user interface to



GENIUS WITHOUT PERSPIRATION

Could sketch recognition cut down the time it takes for the next brilliant innovation to come our way?

Thomas Edison once said, "Genius is one per cent inspiration, 99 per cent perspiration". He was right. His style of inventing demanded that he start with a basic idea and work through all the permutations he could think of, to find the one that worked best. Though this approach may have frustrated some of his employees, there's no doubt that it gave us an astonishing number of inventions. James Dyson, inventor of the famous vacuum cleaner, said that perfecting the cyclone principle by which his invention works took a considerable number of prototypes.

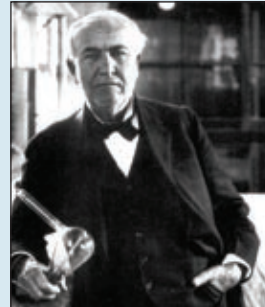
Sometimes, rather than deliberately searching for a solution to a known problem, it takes time for someone to think in the right direction before making a breakthrough. It took 60 years from the invention of the Newcomen steam engine in 1712 for James Watt to think of the second cylinder that reduced its fuel needs by 75 per cent.

The rate of progress at which invention proceeds is limited by the need to find out what will work and what will not. Traditionally, this requires the building of prototypes. But what if inventors could simply sketch their idea on the screen of a tablet PC, note down the real world laws that would govern it and watch animated

cogs, springs and pistons whirl into life? By the time they were ready to build a prototype, they'd already be assured of success.

One can only imagine what people such as Leonardo da Vinci would have made of intelligent sketching. His famous notebooks included masses of inventions as wide ranging as submarines, siege engines and flying machines. Modern-day engineers have attempted to build some of his devices, with only partial success. Because of the time he was working in, much of his work was based on scant knowledge of materials physics. Despite that, we know da Vinci attempted to sell his inventions to noblemen in exchange for patronage. Being able to test a virtual prototype of his giant crossbow, for example, would have shown him that the arms designed to store the energy to launch the bolt at an enemy castle were too thin to take the forces placed on them.

People who class themselves as non-inventors but who have had a flash of inspiration may now be able to see if their idea would work, without



The one per cent genius was there, but Edison could have been helped by sketch recognition



What would a mind such as Leonardo Da Vinci's have made of intelligent sketching?

spending time and money on learning or buying in the skills needed to make a prototype. They could simply sketch it out to see if it worked. That isn't to say that intelligent sketching will turn everyone into inventive geniuses. For most of us, the one per cent inspiration still might not be there. But the lucky few with ideas that could change civilisation will be able to forget the 99 per cent perspiration and engineering skills. Ideas can always come first, not the skills needed to make them reality.

the search function supplied with the interactive Great Buildings Collection CD-ROM. To find a building in the collection, you don't have to be able to spell 'Guggenheim' or 'Acropolis'. You simply have to draw a rough sketch of the building you are looking for, and the software matches the shapes contained within your sketch to the shapes it has found previously in the pictures of the individual buildings contained on the CD.

GRAND DESIGNS

As useful as this ability to understand shapes is for sketching ideas and searching a picture database, engineers still need to get from a sketch to an engineering drawing. In 1998, Hod Lipson, working at the Laboratory for Computer Graphics and Computer Aided Design at Israel's Institute of Technology in Haifa, presented his PhD thesis on turning sketches of 3D objects into proper engineering views. Visual thinking is essential in engineering, and rough sketches are part of the earliest stages of that process. To capture alternative

designs and ideas efficiently, it's necessary to use a computer-aided design (CAD) system as early as possible. It's very easy for experienced engineers to evaluate ideas in terms of sketches. They can quickly assess even 3D objects they've never seen before. But it's difficult to get CAD software to do the same.

The program Lipson wrote as part of his thesis consisted of two parts. The first examined the pen strokes making up the 3D object to discover where the edges lay. From this, it could identify different surfaces and their relationship to each other. It used this information to extract a 3D representation of what started as a 2D sketch of the object. The second part of the system took the 3D internal representation generated from the sketch, decomposed it into faces and edges, and worked out how to produce it from sheet metal. The flat designs produced by the software resembled unglued and unfolded cereal packets that could be used as a plan to create a physical version of the original sketch.

The system then displayed its results, including numeric data about the amount of metal needed to construct the shape, where to cut or bend it to create the originally sketched shape, alternative designs for the user's consideration and even an estimate of the accuracy of its analysis. Rather than present its analysis as an engineering diagram, Lipson decided to give his software the option to sketch the output in the style of the original, reasoning that it would be more acceptable to an engineer used to looking at sketches.

ASSISTANCE REQUIRED

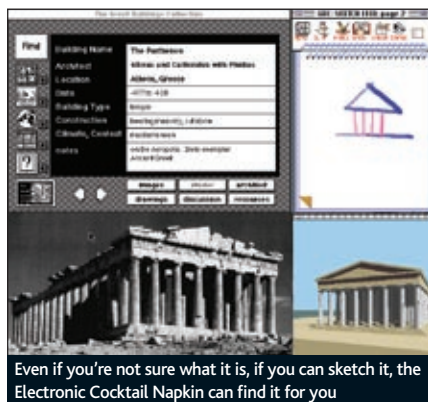
So it was now possible to have software understand the meaning of a sketch, to the point of deconstructing 3D shapes and suggesting how they

might be built. But what about sketching a machine and having the computer animate the paper? In 2003, Professor Randall Davis of the AI Lab at the Massachusetts Institute of Technology showed off the result of his group's work in an online video clip. Called ASSIST, its goal was to allow a designer or engineer to interact with a sketch as if it were a real, mechanical system.

In the clip, Davis drew a slope, like a triangular wedge of cheese, on an electronic whiteboard. At the top of the slope, he drew a child's representation of a flat bed trolley: a large rectangle with a circle under either end. He drew an arrow downward to suggest the direction of gravity, then pressed a button marked 'run'. The car seemed to come to life and started to move, gradually gathered speed under the influence of gravity and finally fell off the end of the slope and off the bottom of the screen. Next he drew a little speed ramp on the slope. This time, the trolley barely had the energy to clear the obstacle before continuing on its way.

Later in the clip, Davis drew a number of similar slopes down the screen. At the end of the lowest slope, he added a U-shaped basket suspended by two spirals, representing springs. At the top of the uppermost slope, he sketched a line of marbles. Pressing run sent them cascading down from slope to slope, finally hitting the side of the basket, whose supporting springs made it wobble back and forth under the bombardment. Davis then halted the run and, after adjusting the position of the slopes, a second run saw all the marbles fall into the basket, whose spring supports lengthened in response to the extra weight.

Davis sees sketching as a more natural way of interacting with computers and thinks it may become dominant. "People will be freed from the



Even if you're not sure what it is, if you can sketch it, the Electronic Cocktail Napkin can find it for you



absurd requirement that we communicate with software by typing and using a mouse. We don't communicate with one another this way, so why should we do so with software?" he asked. If you would like to see an online copy of Davis' demonstration, visit the link at www.ai.mit.edu/projects/rationale/video/davis-v1a.wmv. It's a 20MB file, but it's well worth the wait to see what the software can do.

ASSIST monitors what the user is drawing on an electronic whiteboard and allocates probabilities to various interpretations of what the shapes drawn might represent. The more detail the user draws,

the more the software adjusts these probabilities until it's happy it knows what the sketch means. To do this, it uses a technique from probability theory known as Bayesian analysis, which attempts to link probable causes with observed effects. This and other MIT sketch-recognition projects belong to the Design Rationale Project Group (www.ai.mit.edu/projects/rationale). Underlying some of these innovations is a language called LADDER, designed to describe how the user draws each object within a sketch.

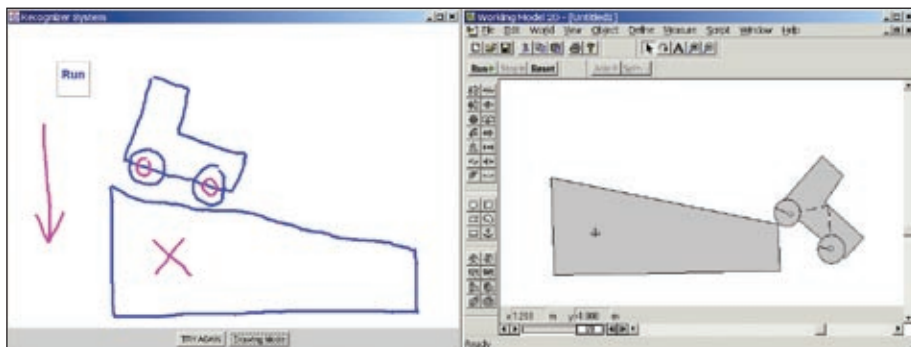
From this, other programs can also work out what sketches mean, how the objects they contain

will interact with each other and how forces of nature should work on the overall sketch. The definitions are detailed. Take a simple open arrowhead. The definition for this three-line object is represented in 33 lines of LADDER, as shown in our Climbing the LADDER box opposite.

SKETCH PAD

Not surprisingly, Microsoft is keen on innovative, sketch-based software for use on the flagging Tablet PC platform. At the Microsoft Professional Developers Conference held in October 2003 in Los Angeles, Microsoft Research senior vice-president Rick Rashid announced that it was actively researching new ways of communicating ideas to computers. One impressive demonstration of what is possible has since become a classic. It concerns an application called MathPad, which was written at Brown University (<http://tinyurl.com/4mdth>) by postgraduate student Joe LaViola.

MathPad allows the user to draw a diagram and associate with it equations that control how different aspects behave. It then creates graphical sequences from these associations. Microsoft has been using the new version, MathPad2, to demonstrate the power of Windows XP Tablet Edition 2005. A demonstrator draws a spring held as a pendulum by a weight attached to its lower



Given the minimum of information in a sketch, MIT's ASSIST software can animate it to act like a group of real objects

THE ELECTRONIC CHALK FACE

How whiteboards could simplify the path from problem to solution

The use of blackboards meant that classrooms were chalky places. Traditional blackboards are great for teaching a whole class or even a lecture theatre, but once you've cleaned them the information they contain is lost forever. In the 1980s, blackboards evolved from being matt to white gloss, and from requiring chalk to marker pens, but there remained the problem of saving work before erasing it and reusing the board.

In the 1990s, with the cheap availability of several technologies, a number of advances were made possible. First came the concept of the computer-controlled projector, taking the video signal from a PC and displaying it on a screen or wall. Presentations could remain on the computer and even contain animation, rather than having to be printed on celluloid sheets or photographic negatives and projected by hand.

As it was being used to display a computer's video output, it made sense to have the screen act as an input device, too, analogous to pen input on a handheld computer. Several technologies have appeared that make this possible. The first is the touch-sensitive surface, of which there are two main types. One type has two sets of electrically conductive contacts, separated by a thin layer of air. When the user presses on the board, it can work out what to send back to the PC based on where a short circuit occurs. The user can use a stylus or a finger to draw on screens of this type. A variation on this principle uses electromagnetism. Electrified wires induce a small current in a coil in the stylus that can be picked up again.

A more flexible approach is to use laser or ultrasonic detection of the user's actions. There

are two basic ways to do this. The first has a laser set at the top left and right corners of the whiteboard. These each send out a beam of light. Associated detectors measure the scattering of the beams by a special reflective stylus, and so can work out its position. The second approach uses ultrasound or infrared signals, emitted by the stylus, and picked up by two detectors. In theory, the detectors could be placed on any flat surface rather than a dedicated whiteboard – the only limitation being the need for a special stylus.

The whiteboard itself connects to a PC. To display what's being drawn, rather than marking the surface of the whiteboard directly, the computer may also be connected to a digital projector. It will be positioned to shine on to the whiteboard, so a

digital representation of what's being drawn on it is overlaid instantly.

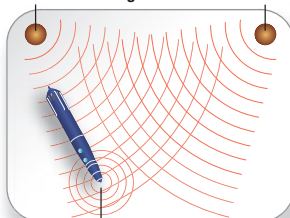
There are several advantages to using an electronic whiteboard. In the classroom, a teacher might re-use notes from previous lessons that illustrate a point particularly well. In a more technologically enabled environment, pupils can sketch out the answers to problems at their own desks, and have the teacher's system display the results for the rest of the class to share. Because the board contains an element of projection as well as being an input device, it can even display educational DVDs and courseware.

In a research environment, not having a spare part of the blackboard to write down flashes of insight is now a thing of the past. Parts of whatever is written can be wiped away without being committed to memory. The logical path from problem to solution can be played out repeatedly to explain to others. It can also be emailed to colleagues for better collaboration.

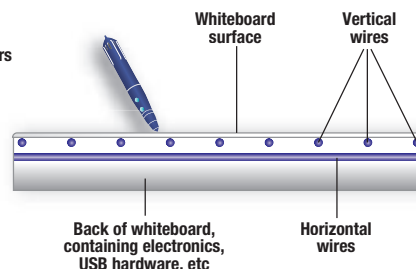
HOW WHITEBOARDS WORK

Some whiteboards send out laser or ultrasonic signals to map the position of a stylus or even the user's finger

In some designs, sensors at the corners detect a signal given out by the stylus



Some styluses give out a signal, either powered by onboard batteries, or induced using an onboard coil and a current flowing through the board



An electronic whiteboard surface seen edge on. In some designs, wires carry a current that induces electricity in a coil powering the stylus. In other designs, the stylus presses against the lattice of wires, shorting them out at a specific junction



CLIMBING THE LADDER

How the LADDER language is helping computers understand and develop preliminary sketches

Underlying the pen-based interface work at MIT's Design Rationale Group is a language called LADDER, used to describe how onscreen shapes are drawn, what they consist of, how to display them, how they may be edited and more.

The language consists of built-in shapes such as points, lines, circles, ellipses, boxes and spirals. It also contains a special syntax for describing how the basic shapes fit together to form far more complex objects. It also contains statements to extend the language, so it could describe practically anything the user sketches out.

Because of the number of ways there are to draw out even the most basic of shapes, LADDER descriptions can quickly become very detailed indeed. Let's take as an example a simple open arrow, consisting of three lines. The result of drawing such a shape on the screen might generate LADDER statements as follows:

```
(define shape OpenArrow
  (description "An arrow with an open
  head"))
  (components
    (Line shaft)
    (Line head1)
```

```
(Line head2)))
(constraints
  (coincident shaft.p1 head1.p1)
  (coincident shaft.p1 head2.p1)
  (coincident head1.p1 head2.p1)
  (equal-length head1 head2)
  (acute-meet head1 shaft)
  (acute-meet shaft head2))
  (aliases
    (Point head shaft.p1)
    (Point tail shaft.p2))
  (editing
    ( (trigger (click_hold_drag shaft))
      (action
        (translate this)
        (set-cursor DRAG)
        (show-handle MOVE tail head))
      ( (trigger (click_hold_drag head))
        (action
          (rubber-band this head tail)
          (show-handle MOVE head)
          (set-cursor DRAG))
        ( (trigger (click_hold_drag tail))
          (action
            (rubber-band this tail head)
            (show-handle MOVE tail))
```

```
(set-cursor DRAG)))
(display (original-strokes)))
```

LADDER is advanced enough to describe dashed lines and those having multiple segments. In developing the language, MIT's Tracy Hammond and Randal Davis wanted to capture the ways in which people describe simple shapes, so as not to make the language too obscure. To ensure this, they asked 35 people to describe a library of simple shapes in everyday English. Their aim is to produce a language in which users could even describe complex shapes if needs be.

A powerful expert system called Jess generates language statements in response to what the user draws in the sketching programs. This takes the items drawn onscreen and attempts to find what classes of shape they contain. From this it can generate LADDER definitions and even add new rules when the user describes unique shapes or combines existing ones into more complex sketches. The hope is that because LADDER is designed specifically to handle sketches in general rather than in closely defined application areas, it will become a standard for the description and interchange of sketched ideas.

end. He also sketches out a few equations, presses run, and the spring and weight bounce up and down, back and forth just as they would in real life. Slowly, the motion of spring and weight grinds to a halt. They are, in fact, acting entirely under the control of the sketched equations and their starting positions within the diagram. A quick pen gesture, and a graph of the equations over time appears, revealing the fading rhythmic forces acting on the spring and weight.

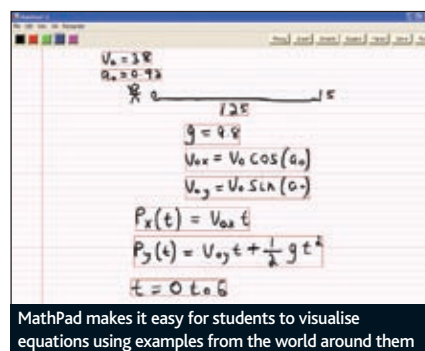
"Microsoft Research believes that the student notebook of tomorrow is really this highly sophisticated, digital mobile device," said Microsoft technology evangelist John San Giovanni. IBM spokesperson Harriet Ip agreed. "The evolution of e-business is pushing computing beyond traditional PCs, beyond wireless phones and PDAs, to a new generation of devices," she said.

To use MathPad, the user first draws a diagram, which the software splits into its component parts and analyses for later use. Next the user draws out a number of equations, associates them with a letter and draws the letter on the part of the diagram to be controlled by the equation. So, for instance, one might draw two flatbed trolleys, associate one with an equation for constant velocity and the other with an equation describing acceleration from a standing start. Pressing run would show one travelling as the other gathers speed. The hope is that using MathPad to visualise equations will lead to a deeper understanding of physics and mathematics than is possible by studying dry equations alone.

The user communicates with MathPad entirely through a combination of handwriting recognition and graphical gestures. The diagrams drawn are compared against a library of primitive shapes, just as in earlier systems. The user writes out equations

and lassoes them to invoke the recognition algorithm. He can then label them and add the label to any part of the diagram that the equation will control. Because the heart of the system is the ability to show how equations affect the behaviour of physical systems over time, it allows engineers and students alike to show in a dramatic way whether something is possible without having to perform an experiment.

An example from the MathPad website shows a baseball player attempting to hit a pitch out of a stadium, along with equations describing the laws of motion under gravity. MathPad makes it easy to show just how much force is required for the ball to leave the stadium, from which it's possible to work out whether a human can perform such a feat. Running the diagram immediately shows if there is a mistake in the equations, too, making it a potentially useful teaching tool. A plus rather than a minus sign in an equation may make the ball shoot up into space rather than down towards the ground. From the equation itself, it's sometimes hard to see such mistakes, but on an animated diagram it's obvious.



The tablet PC, which has low European sales, needs a killer application and many pundits blame Microsoft for not supplying one. But the sketch-based interface might be that application, as Lipson's new research further indicates.

THE BIG SCREEN

Since becoming a professor at the Computational Synthesis Lab at Cornell University in the US, Lipson has been writing CAD systems that work entirely in sketches, removing the transition from sketch to CAD. His 3D Journal project (<http://tinyurl.com/6fu8q>) is the latest example.

3D Journal can understand complex sketches drawn freehand on the screen of a tablet PC well enough to realise what it must do to turn 2D representations drawn on a flat screen into virtual 3D objects, capable of being angled and rotated. Lipson hopes to be able to test these objects using simulated real-world physics, thereby blurring the boundaries between ideas and prototypes and speeding product development in the process. Written using Microsoft's .NET framework with the Pen SDK, the latest version is available for free download to XP Tablet Edition PCs.

Colleagues in other Cornell departments are also working on 3D printing technology. The sum total of current work could take engineers from sketch to prototype without compromising the way they express themselves. But where else might sketch recognition take us in the near future? One distinct possibility is a marriage with evolutionary computing. Genetic algorithms are widely used to design everything from turbo fan blades to electronic circuits. Allowing software to evolve interesting alternatives from initial sketches could unleash creative power in ways that have been unavailable to us until now. **CS**